IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Patent Application

Inventors: Peter Joseph Giacomini et al.

Serial No.: 09/725732

Filing Date: 11/29/2000

Art Unit: 2142

Examiner: Thong H Vu

Docket No.: 500-001US

Title: Distributed Caching Architecture For Computer Networks

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

APPEAL BRIEF UNDER 37 CFR 41.67

Pursuant to 37 CFR 41.67, this brief is filed in support of the appeal in this application.

REAL PARTY IN INTEREST

The real party of interest in this application is the assignee of this application: Broadspider Networks, Inc., formerly of Shrewsbury, New Jersey. The attorneys for this Appeal, DeMont & Breyer, LLC, have an equity interest in the assignee of this application.

RELATED APPEALS AND INTERFERENCES

There are no related appeals or interferences.

STATUS OF CLAIMS

Claims 1-22 stand rejected and are being appealed.

STATUS OF AMENDMENTS

All amendments have been entered.

SUMMARY OF THE CLAIMED SUBJECT MATTER

To make it easier for the average reader to fully understand and appreciate the present invention, the present invention is explained as it might be applied in the context of the World Wide Web. The scope of the claims, however, is not so limited.

Prior Art

When a user of the World Wide Web requests a Web page, the user must wait until the page is available on his or her client (*e.g.*, computer, *etc.*) for viewing. In general, this wait occurs because the request for the Web page and the name of the Web page must traverse the Internet from the client to the Web server that is the source of the page, the request must be fulfilled, and the requested page must travel back to the client. If the Internet is congested or the Web server that is the source of the page is overwhelmed with many concurrent requests for pages, the wait can be considerably long. (Applicants Specification Paragraph [0003])

To shorten this wait, caches are deployed throughout the Internet that store commonly-requested Web pages. The presence of caches in the Internet expedites the delivery of Web pages to clients in two ways. First, a cache eliminates the need for the request to travel all of the way to the Web server, and, therefore, eliminates some the wait associated with the transit. Second, a cache also reduces the number of Web page requests that must be fulfilled by the Web server, and, therefore, reduces the wait associated with contention for the Web server. (Applicants Specification Paragraph [0004])

The information that a cache stores, and the mechanism that it uses to index that information is unintuitive. In order to understand this, let's talk about how a typical paper telephone book works.

A "normal" telephone book provides a mapping of names to telephone numbers and is said to be indexed by name, as shown in Table 1.

Name	Telephone Number
Orange, William	545-223-4342
Rehnquist, William	103-554-2232
Schweitzer, Albert	222-555-1212
Taylor, Zachary	212-443-4523
Tell, William	344-244-4444

Table 1 — Telephone Directory Indexed by Name

In other words, given a name, you can quickly and easily find the telephone number associated with the name. Given a telephone number, you could find the name associated with the telephone number in a normal telephone book, but it would be slow.

A "reverse" telephone book is indexed by telephone number, as shown in Table 2.

Telephone Number	Name
103-554-2232	Rehnquist, William
212-443-4523	Taylor, Zachary
222-555-1212	Schweitzer, Albert
344-244-4444	Tell, William
545-223-4342	Orange, William

Table 2 — Telephone Directory Indexed by Telephone Number

In other words, given a telephone number, you can quickly and easily find the name associated with the telephone number. Here, given a name you could find the telephone number associated with the name, but it would be slow.

Here comes the unintuitive part. You might think that a cache of Web pages is indexed by name like a normal telephone book, but it is not. Why? Because the names of Web pages have different lengths, as measured in bits, which leads to computational inefficiency in the cache. A typical cache almost always runs nears its limit, and, therefore, anything that can be done to improve this efficiency is advantageous.

To prevent this inefficiency, a cache of Web pages is indexed by a string that is a function or "nickname" of the name of the Web page. The telephone directory in Table 1 indexed by nickname appears in Table 3.

Nickname	Telephone Number
Al	222-555-1212
Bill	545-223-4342
Bill	103-554-2232
Bill	344-244-4444
Zeke	212-443-4523

Table 3 — Telephone Directory Indexed by Nickname

A problem with nicknames is that many different people often have the same nickname and it isn't possible to distinguish William Rehnquist's telephone number from William of Orange's or William Tell's in the directory in Table 3. To ensure that the cache of Web pages does not suffer the same problem, *the cache stores both the Web page and the*

name of the Web page together and the combination is indexed by the nickname string. This enables the cache to distinguish or "disambiguate" between Web pages whose names have the same nickname.

A telephone directory like this appears in Table 4.

Nickname	Name	Telephone Number
Al	Schweitzer, Albert	222-555-1212
Bill	Orange, William	545-223-4342
Bill	Rehnquist, William	103-554-2232
Bill	Tell, William	344-244-4444
Zeke	Taylor, Zachary	212-443-4523

Table 4 — Telephone Directory Indexed by Nickname and Disambiguated by Name

William Rehnquist's telephone number can be obtained from the telephone directory in Table 4 first by computing his nickname "Bill" and then by using his full name to disambiguate among the three entries for Bill.

Although caches for Web pages use this mechanism, the process of disambiguation takes time. Therefore, a function that reduces the likelihood that the same nickname string will be generated from two different Web page names is preferred. Such functions have unusual mathematical properties and are known as "hash functions." The "nickname" strings that hash functions generate are called "hash keys."

Now it is time to see how all of the works together. In the prior art, when a user of a client machine desires a Web page, the name or "URL" of the Web page traverses the Internet from the client machine to the cache of Web pages. (It is not germane to the present invention what pages are stored in the cache or how they got there.) The cache uses the hash function on the name of the Web page to generate the hash key or nickname of the Web page. The cache then uses the hash key as the index into its database to find the Web page. If there is more than one entry in the database with the same name the cache uses the name of the Web page to disambiguate among the entries. The cache then sends the proper Web page back to the user's client for viewing.

The Invention

The inventors of the present invention recognized that caches in the prior art were spending too many computational resources performing the hash and disambiguation functions and that those functions could be more efficiently handled by the user's client machine.

In accordance with one illustrative embodiment of the present invention, when a user of a client machine desires a Web page, the user types in or selects the name of the desired Web page. *The user's client performs the hash function of the name of the Web page* and generates the hash key or nickname of the Web page. The user's client transmits:

i. the hash key or nickname of the Web page, and

ii. a request for the Web page

to the cache. The cache receives the hash key and the request for the Web page and uses the hash key as the index into its database. If the database contains more than one entry for that nickname the cache retrieves the name of the Web page from the database. The cache then transmits the name of the Web to the client and asks the client if that is the desired Web page. The client compares the name of the desired Web page to the name of the Web page received from the cache. If there is a match, the client directs the cache to send the Web page. If there is no match, the client directs the cache to try again. In this way the client receives the Web page and reduces the computational burden on the cache. This enables the cache to handle more requests per unit time than in the prior art.

In accordance with a second illustrative embodiment of the present invention, the user's client performs the hash function and transmits:

i. the hash key or nickname of the Web page,

ii. the name of the Web page, and

iii. a request for the Web page

to the cache. This alleviates the cache from the burden of computing the hash function while allowing the cache to perform the disambiguation function. This is also advantageous in that it relieves the cache of the burden of computing the hash function and eliminates the necessity for the cache and the client to cooperate in the disambiguation process.

GROUNDS OF OBJECTION AND REJECTION TO BE REVIEWED ON APPEAL

Ground 1: Rejection of Claim 15

Claim 15 was rejected under 35 U.S.C. 101 because "the claimed invention lacks patentable utility.

Ground 2: 35 U.S.C. 112 Rejection of Claim 16

Claim 16 was rejected under 35 U.S.C. 112, Second Paragraph, for failing to particularly point out and distinctly claim the subject matter which the applicant regards as the invention.

Ground 3: 35 U.S.C. 102 Rejection of Claims 1-20

Claims 1-20 were rejected under 35 U.S.C. 102(b) as being anticipated by P.R. Carpentier, U.S. Patent 6,807,632 (hereinafter "Carpentier").

Ground 4: 35 U.S.C. 103 Rejection of Claims 21 and 22

Claims 21 & 22 were rejected under 35 U.S.C. 103(a) as being unpatentable over P.R. Carpentier, U.S. Patent 6,807,632 (hereinafter "Carpentier") in view of J. Takahashi, U.S. Patent 5,428,774 (hereinafter "Takahashi").

ARGUMENTS

Ground 1: Rejection of Claim 15

Claim 15 has been rejected under 35 U.S.C. 101 because the claimed invention lacks "patentable utility." The applicants respectfully traverse.

Claim 15 recites:

> **15.** (Original) A method comprising:
>
> *receiving* a request for a first resource and a hash key that is a hashed function of a first resource identifier;
>
> *retrieving* said first resource and said first resource identifier from a data structure that is indexed by said hash key; and
>
> *transmitting* said first resource and said first resource identifier in response to said request for said first resource.
>
> (*emphasis supplied*)

Claim 15 recites the receiving, retrieving, and transmission of various items, and, as such, achieves a "useful, concrete, and tangible result" in accordance with the Patent Office's Interim Guidelines for Examiner of Patent Applications for Patent Subject Matter Eligibility. This is true whether the request, hash key, resource identifiers, and resources are, for example, electromagnetic signals, papers, or papyrus scrolls. For this reason, the applicants respectfully submit that claim 15 recites patentable subject matter within the requirement of 35 USC 101 and that the rejection is traversed.

Ground 2: 35 U.S.C. 112 Rejection of Claim 16

Independent claim 16 was rejected under 35 U.S.C. 112, Second Paragraph, for failing to particularly point out and distinctly claim the subject matter which the applicant regards as the invention. The applicants respectfully traverse the rejection.

Claim 16 recites:

> **16.** (Original) An apparatus comprising:
>
> a receiver for receiving a request for a first resource and a hash key that is a hashed function of a first resource identifier;
>
> a processor for retrieving said first resource and said first resource identifier from a data structure that is indexed by said hash key; and
>
> a transmitter for transmitting said first resource and said first resource identifier in response to said request for said first resource.
>
> (*emphasis supplied*)

There is no vagueness or ambiguity of the scope of claim 16, and, therefore, the applicants respectfully submit that the rejection of claim 16 is traversed.

Ground 3: 35 U.S.C. 102 Rejection of Claims 1-20

Claims 1-20 were rejected under 35 U.S.C. 102(b) as being anticipated by P.R. Carpentier, U.S. Patent 6,807,632 (hereinafter "Carpentier"). The applicants respectfully traverse.

Claim 1 recites:

> **1.** (Original) A method comprising:
>
> *hashing at a first processor a first resource identifier to create a hash key, wherein said first resource identifier identifies a first resource*;
>
> transmitting from said first processor to a second processor said hash key and a request for said first resource; and
>
> receiving at said first processor a second resource in response to the transmission of said hash key and said request for said first resource.
>
> (*emphasis added*)

Nowhere does Carpentier teach or suggest, alone or in combination with the other references, what claim 1 recites — namely, hashing the "name" of the resource to create a "nickname" of the resource.

Carpentier teaches that the more or less arbitrary naming of files that we all use today or what he calls "potentially misleading extrinsic naming" leads to several problems on computer systems. These include:

(1) How do you know when two files with the same name are identical or not?

(2) How do you know when two files with different names are identical or not?

(3) How do you prevent duplicate copies of the same file from taking resources when it is not necessary?

Carpentier's answer is to replace the "normal" name that files are typically given with a digital signature or fingerprint for the file, which is based on the contents of that file. Done properly, this virtually ensures that:

(1) Two files with identical fingerprints are identical, and

(2) Two files with different fingerprints are different.

As those skilled in the art would know, the answer is to use a hash function of the file to generate the fingerprint.

The result is that when Carpentier wants a file from a remote system, which indexes the files based on their digital fingerprint, Carpentier transmits the digital fingerprint of the file to the remote system alone with a request for the file. As advantageous as the teaching of Carpentier might be, nowhere does it teach or suggest, alone or in combination with the other references, what is recited in claim 1. For these reasons, the applicants respectfully submit that the rejection of claim 1 is traversed.

Because claims 2-7 depend on claim 1, the applicants respectfully submit that the rejection of them is also traversed.

Independent claim 8 recites:

> **8.** (Original) An apparatus comprising:
>
> a first processor for *hashing a first resource identifier to create a hash key, wherein said first resource identifier identifies a first resource*;
>
> a transmitter for transmitting said hash key and a request for said first resource to a second processor; and
>
> a receiver for receiving a second resource in response to the transmission of said hash key and said request for said first resource.
>
> (*emphasis supplied*)

Nowhere does Carpentier teach or suggest, alone or in combination with the other references, what claim 8 recites — namely, hashing the "name" of the resource to create a "nickname" of the resource. For these reasons, the applicants respectfully submit that the rejection of claim 8 is traversed.

Because claims 9-14 depend on claim 8, the applicants respectfully submit that the rejection of them is also traversed.

Independent claim 15 recites:

> **15.** (Original) A method comprising:
>
> receiving a request for a first resource and *a hash key that is a hashed function of a first resource identifier*;
>
> retrieving said first resource and said first resource identifier from a data structure that is indexed by said hash key; and
>
> transmitting said first resource and said first resource identifier in response to said request for said first resource.
>
> (*emphasis supplied*)

Nowhere does Carpentier teach or suggest, alone or in combination with the other references, what claim 15 recites — namely, hashing the "name" of the resource to create a "nickname" of the resource. For these reasons, the applicants respectfully submit that the rejection of claim 15 is traversed.

Independent claim 16 recites:

> **16.** (Original) An apparatus comprising:
>
> a receiver for receiving a request for a first resource and *a hash key that is a hashed function of a first resource identifier;*
>
> a processor for retrieving said first resource and said first resource identifier from a data structure that is indexed by said hash key; and
>
> a transmitter for transmitting said first resource and said first resource identifier in response to said request for said first resource.
>
> (*emphasis supplied*)

Nowhere does Carpentier teach or suggest, alone or in combination with the other references, what claim 16 recites — namely, hashing the "name" of the resource to create a "nickname" of the resource. For these reasons, the applicants respectfully submit that the rejection of claim 16 is traversed.

Independent claim 17 recites:

> **17.** (currently amended) A method comprising:
>
> receiving from a first processor a first resource identifier that identifies a first resource, *a hash key that is a hashed function of said first resource identifier*, and a request for a first resource;
>
> retrieving a second resource and a second resource identifier from a data structure that is indexed by said hash key;
>
> verifying that said second resource is said first resource by comparing said second resource identifier to said first resource identifier; and
>
> transmitting said second resource to said first processor when said second resource is verified as said first resource.
>
> (*emphasis supplied*)

Nowhere does Carpentier teach or suggest, alone or in combination with the other references, what claim 17 recites — namely, hashing the "name" of the resource to create a "nickname" of the resource. For these reasons, the applicants respectfully submit that the rejection of claim 17 is traversed.

Independent claim 18 recites:

> **18.** (previously presented) An apparatus comprising:
>
> a receiver for receiving from a first processor a first resource identifier that identifies a first resource, *a hash key that is a hashed function of said first resource identifier*, and a request for a first resource;
>
> a second processor for retrieving a second resource and a second resource identifier from a data structure that is indexed by said hash key, and for verifying that said second resource is said first resource by comparing said second resource identifier to said first resource identifier; and
>
> a transmitter for transmitting said second resource to said first processor when said second resource is verified as said first resource.
>
> (*emphasis supplied*)

Nowhere does Carpentier teach or suggest, alone or in combination with the other references, what claim 18 recites — namely, hashing the "name" of the resource to create a "nickname" of the resource. For these reasons, the applicants respectfully submit that the rejection of claim 18 is traversed.

Independent claim 19 recites:

> **19.** (Original) A method comprising:
>
> hashing at a first processor *a first resource identifier to create a hash key, wherein said first resource identifier identifies a first resource*;
>
> transmitting from said first processor to a second processor said hash key and a request for said first resource when said all or a portion of said hash key is contained in a list of valid hash keys associated with said first processor; and
>
> receiving at said first processor said first resource in response to the transmission of said hash key and said request for said first resource.
>
> (*emphasis supplied*)

Nowhere does Carpentier teach or suggest, alone or in combination with the other references, what claim 19 recites — namely, hashing the "name" of the resource to create a "nickname" of the resource. For these reasons, the applicants respectfully submit that the rejection of claim 19 is traversed.

Independent claim 20 recites:

> **20.** (previously presented) An apparatus comprising:
>
> a first processor for *hashing a first resource identifier to create a hash key*, wherein said first resource identifier identifies a first resource, and for verifying that all or a portion of said hash key is contained in a list of valid hash keys;
>
> a transmitter for transmitting from said first processor to a second processor said hash key and a request for said first resource; and
>
> a receiver for receiving said first resource in response to the transmission of said hash key and said request for said first resource.
>
> (*emphasis supplied*)

Nowhere does Carpentier teach or suggest, alone or in combination with the other references, what claim 20 recites — namely, hashing the "name" of the resource to create a "nickname" of the resource. For these reasons, the applicants respectfully submit that the rejection of claim 20 is traversed.

Ground 4: 35 U.S.C. 103 Rejection of Claims 21 and 22

Claims 21 & 22 were rejected under 35 U.S.C. 103(a) as being unpatentable over P.R. Carpentier, U.S. Patent 6,807,632 (hereinafter "Carpentier") in view of J. Takahashi, U.S. Patent 5,428,774 (hereinafter "Takahashi"). The applicants traverse.

Independent claim 21 recites:

> **21.** (previously presented) A method comprising:
>
> receiving from a first processor a request for a first resource and *a first hash key that is a hashed function of a first resource identifier*;
>
> retrieving a second resource and a first portion of a second hash key from a data structure that is indexed by a first portion of said first hash key;
>
> verifying that said second resource is said first resource by comparing a second portion of said first hash key to said first portion of said second hash key; and
>
> transmitting said second resource to said first processor when said second resource is verified as said first resource.
>
> (*emphasis supplied*)

Nowhere does Carpentier or Takahashi teach or suggest, alone or in combination with the other references, what claim 21 recites — namely, hashing the "name" of the resource to create a "nickname" of the resource. For these reasons, the applicants respectfully submit that the rejection of claim 21 is traversed.

Independent claim 22 recites:

> **22.** (previously presented) An apparatus comprising:
>
> a receiver for receiving from a first processor a request for a first resource and a first hash key that is a hashed function of a first resource identifier;
>
> a second processor for retrieving a second resource and a first portion of a second hash key from a data structure that is indexed by a first portion of said first hash key, and for verifying that said second resource is said first resource by comparing a second portion of said first hash key to said first portion of said second hash key; and
>
> a transmitter for transmitting said second resource to said first processor when said second resource is verified as said first resource.
>
> (*emphasis supplied*)

Nowhere does Carpentier or Takahashi teach or suggest, alone or in combination with the other references, what claim 22 recites — namely, hashing the "name" of the

resource to create a "nickname" of the resource. For these reasons, the applicants respectfully submit that the rejection of claim 22 is traversed.

CONCLUSION

The applicants have demonstrated that the logic underlying the Office's rejection is untenable, and, therefore, that the rejection is not sustainable. For this reason, the applicants respectfully request the Board of Appeals to reverse the decision of the Examiner as provided for in 37 C.F.R. 41.50(a).

Respectfully,
Peter Joseph Giacomini et al.

By **/Jason Paul DeMont/**
Jason Paul DeMont
Reg. No. 35793
Attorney for Applicants
732-578-0103 x11

DeMont & Breyer, L.L.C.
Suite 250
100 Commons Way
Holmdel, NJ 07733
United States of America

Claims Appendix

1. (Original) A method comprising:

hashing at a first processor a first resource identifier to create a hash key, wherein said first resource identifier identifies a first resource;

transmitting from said first processor to a second processor said hash key and a request for said first resource; and

receiving at said first processor a second resource in response to the transmission of said hash key and said request for said first resource.

2. (Original) The method of claim 1 further comprising receiving at said first processor a second resource identifier in response to the transmission of said hash key and said request for said first resource.

3. (Original) The method of claim 2 wherein said first processor verifies that said second resource is said first resource by comparing said second resource identifier to said first resource identifier.

4. (Original) The method of claim 1 further comprising transmitting from said first processor to said second processor said first resource identifier in addition to said hash key and said request for said first resource.

5. (Original) The method of claim 4 wherein said second processor stores said second resource and said second resource identifier in a data structure that is indexed by said hash key.

6. (Original) The method of claim 5 wherein said second processor verifies that said second resource is said first resource by comparing said second resource identifier to said first resource identifier.

7. (Original) The method of claim 1 wherein said hash key and said request for said first resource are transmitted from said first processor to said second processor when said all or a portion of said hash key is contained in a list of valid hash keys associated with said first processor.

8. (Original) An apparatus comprising:

a first processor for hashing a first resource identifier to create a hash key, wherein said first resource identifier identifies a first resource;

a transmitter for transmitting said hash key and a request for said first resource to a second processor; and

a receiver for receiving a second resource in response to the transmission of said hash key and said request for said first resource.

9. (Original) The apparatus of claim 8 wherein said receiver also receives a second resource identifier in response to the transmission of said hash key and said request for said first resource.

10. (Original) The apparatus of claim 9 wherein said first processor verifies that said second resource is said first resource by comparing said second resource identifier to said first resource identifier.

11. (Original) The apparatus of claim 8 wherein said transmitter also transmits to said second processor said first resource identifier in addition to said hash key and said request for said first resource.

12. (Original) The apparatus of claim 11 wherein said second processor stores said second resource and said second resource identifier in a data structure that is indexed by said hash key.

13. (Original) The apparatus of claim 12 wherein said second processor verifies that said second resource is said first resource by comparing said second resource identifier to said first resource identifier.

14. (Original) The apparatus of claim 8 wherein said hash key and said request for said first resource are transmitted from said first processor to said second processor when said all or a portion of said hash key is contained in a list of valid hash keys associated with said first processor.

15. (Original) A method comprising:

receiving a request for a first resource and a hash key that is a hashed function of a first resource identifier;

retrieving said first resource and said first resource identifier from a data structure that is indexed by said hash key; and

transmitting said first resource and said first resource identifier in response to said request for said first resource.

16. (Original) An apparatus comprising:

a receiver for receiving a request for a first resource and a hash key that is a hashed function of a first resource identifier;

a processor for retrieving said first resource and said first resource identifier from a data structure that is indexed by said hash key; and

a transmitter for transmitting said first resource and said first resource identifier in response to said request for said first resource.

17. (currently amended) A method comprising:

receiving from a first processor a first resource identifier that identifies a first resource, a hash key that is a hashed function of said first resource identifier, and a request for a first resource;

retrieving a second resource and a second resource identifier from a data structure that is indexed by said hash key;

verifying that said second resource is said first resource by comparing said second resource identifier to said first resource identifier; and

transmitting said second resource to said first processor when said second resource is verified as said first resource.

18. (previously presented) An apparatus comprising:

a receiver for receiving from a first processor a first resource identifier that identifies a first resource, a hash key that is a hashed function of said first resource identifier, and a request for a first resource;

a second processor for retrieving a second resource and a second resource identifier from a data structure that is indexed by said hash key, and for verifying that said second resource is said first resource by comparing said second resource identifier to said first resource identifier; and

a transmitter for transmitting said second resource to said first processor when said second resource is verified as said first resource.

19. (Original) A method comprising:

hashing at a first processor a first resource identifier to create a hash key, wherein said first resource identifier identifies a first resource;

transmitting from said first processor to a second processor said hash key and a request for said first resource when said all or a portion of said hash key is contained in a list of valid hash keys associated with said first processor; and

receiving at said first processor said first resource in response to the transmission of said hash key and said request for said first resource.

20. (previously presented) An apparatus comprising:

a first processor for hashing a first resource identifier to create a hash key, wherein said first resource identifier identifies a first resource, and for verifying that all or a portion of said hash key is contained in a list of valid hash keys;

a transmitter for transmitting from said first processor to a second processor said hash key and a request for said first resource; and

a receiver for receiving said first resource in response to the transmission of said hash key and said request for said first resource.

21. (previously presented) A method comprising:

receiving from a first processor a request for a first resource and a first hash key that is a hashed function of a first resource identifier;

retrieving a second resource and a first portion of a second hash key from a data structure that is indexed by a first portion of said first hash key;

verifying that said second resource is said first resource by comparing a second portion of said first hash key to said first portion of said second hash key; and

transmitting said second resource to said first processor when said second resource is verified as said first resource.

22. (previously presented) An apparatus comprising:

a receiver for receiving from a first processor a request for a first resource and a first hash key that is a hashed function of a first resource identifier;

a second processor for retrieving a second resource and a first portion of a second hash key from a data structure that is indexed by a first portion of said first hash key, and for verifying that said second resource is said first resource by comparing a second portion of said first hash key to said first portion of said second hash key; and

a transmitter for transmitting said second resource to said first processor when said second resource is verified as said first resource.

Evidence Appendix

There is no evidence submitted pursuant to 37 CFR §§ 1.130, 1.131, or1.132.

Related Proceedings Appendix

There are no related proceedings.